ورژن node که نصب کرده ایم:

node –version

ورژن npm که نصب کرده ایم:

Npm --version

در سایت خود react یک جایی را معرفی میکند که در انجا میتوان کد ری اکت نوشت :

<http://codepen.io/gaearon/pen/ZpvBNJ?editors=0010>

این خط

const element = <h1>Hello, world!</h1>;

یک خط jsx است . jsx المنت های react را تولید می کند

Webpack یک ماژول است که JSX, React, و دیگر وابسته هایشان را برمیدارد و انهارا برای مرورگر مامپایل می کند. این کار را به این شکل انجام میدهد که تمام کدها و فایل هایی که ما ایجاد کرده ایم را برمیدارد و در یک بسته به یک فایل js یک css و فایل های دیگری که احتمالا هستند بدل می کند.

در react مفهومی به نام کامپوننت وجود دارد . ما باید بتوانیم برای خودمان کامپوننت هایی که میخواهیم را بنویسیم. خط زیر یک کامپوننت است:  
<Hello />

و حالا همین کامپوننت به یک div تبدیل می شود. مثلا میتواند این بشود:

<div>Hello, haters!</div>

اما چگونه یک کامپوننت مثل کامپوننت فوق به یک کد html تبدیل می شود؟ جواب در یکی از متد های react به نام render است این متد هر نوع html که بخواهی را تف می کند.

اگر بخواهیم که react بتواند نود های مارو اپدیت کند دیگر حق نداریم به روض قدیمی نود ایجاد کنیم. بلکه باید با یکی از متدهای react کدهای html یا همان المان های dom خود را ایجاد کنیم برای اینکار متد [createElement](https://facebook.github.io/react/docs/glossary.html" \t "_blank) را در اختیارمان گذاشته اند.

مطالب مفیدی در مورد react در لینک زیر است چند خط فوق را هم از همین صفحه اورده ام:

<https://thinkster.io/tutorials/create-a-react-component>

البته همان طور که وضوح پیداست createelement کدایش گیج و اشبته هستند و ما دوست داریم تا کدهای اج تی ام ال قدیمی را بنویسیم پس از jsx استفاده می کنیم. سایتی هست که میتواند برایمان jsx را کدهای جاوااسکریپتی تبدیل کند. برای دیدن این سایت به ادرس زیر برو:

<http://babeljs.io/repl/#?babili=false&evaluate=true&lineWrap=false&presets=es2015%2Creact%2Cstage-2&targets=&browsers=&builtIns=false&debug=false&experimental=true&code=const%20App%20%3D%20function%28%29{%0A%20%20return%20%3Cdiv%3Ehi%3C%2Fdiv%3E%3B%0A%0A}>

خودش انچه لازم است را به همراه مثال نشانمان میدهد انچه من نوشته بودم این بود:

const App = function(){

return <div>hi</div>;

}

ببین هنوز همین است یا نه

کامپوننت ها با کلمه ی کلیدی ه createClass ایجاد میشوند.

React نام یک شی است و createClass نام یک متد است که میتواند روی این شی اجرا شود ما با ان میتوانیم کامپوننت ایجاد کنیم.

برای کار با react باید دو تا فایل را ایمپورت کنیم این هارا:

React و reactdom

بدیهی است که ابتدا react و سپس reactdom باید ایمپورت شوند زیرا reactdom از react استفاده می کند.

انها را از طریق کدهای زیر میتوان ایپورت کرد:

<script src=" https://fb.me/react-0.14.3.js"></script>

<script src="https://fb.me/react-dom-0.14.3.js"></script>

اما اگر اینکار را بکنیم نیاز ب فیلتر شکن هست پس راه دیگر این هست که :  
از اینجا ایمپورت کنیم که فیلتر هم نیست:

<script src="https://unpkg.com/react@latest/dist/react.js"></script>

<script src="https://unpkg.com/react-dom@latest/dist/react-dom.js"></script>

و راه سوم هم البته دانلود انها و لینک کردن محلی به انها مانند زیر است که البته واضح است که فایل ها رو درون فولدری به نام lib قرار داده ام:

<script src="lib/react-0.14.3.js"></script>

<script src="lib/react-dom-0.14.3.js"></script>

Babel

کار بابل این است که jsx را تبدیل کند.

بابل برای این است که مرورگرهای قدیمی هم که از es5 استفاده میکنند بتوانند از کدهای es6 سردر بیاورند(این حرف هم طبق djangocon صحیح است) برای اشنایی از ان میتوان به ادرس زیر سر زد:

<https://babeljs.io/docs/setup/#installation>

برای استفاده از ان همانگونه که در سایت فوق هم امده تنها کافیست که یکی از ادرس های زیر رو لینک کنیم فعلا که هردو کار میکرده اند. این لینک کردن باید بعد از react و reactDom باشد :

<script src="https://unpkg.com/babel-standalone@6.15.0/babel.min.js"></script>

<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>

بعد از لینک کردن تنها کافیست در قسمت type تگ script به babel اشاره کنیم یعنی کدها درون این قرار میگیرند:

<script type="text/babel">

</script>

یک اموزش پیدا کردم روی سایت خود ری اکت که یک بازی دوز را تولید میکند به نظرم خواندنش خوب است

<https://facebook.github.io/react/tutorial/tutorial.html>

یک سایت خیلی خوب دیگر هم پیدا کردم این کارش از خیلی خوب گذشته دمش گرم:

<http://buildwithreact.com/tutorial/components>

چگونه محیط را اماده کنیم:

<http://blog.tamizhvendan.in/blog/2015/11/23/a-beginner-guide-to-setup-react-dot-js-environment-using-babel-6-and-webpack/>

یک سایت اموزش برای react پیدا کردم که یک مثال to do list خیلی خوب دارد :

<https://blog.alexdevero.com/create-todo-app-react/>

دستور npm init یک فایل package.json برایمان ایجاد میکند و میدانیم که این فایل قرار است اسامی فایل ها یا بهتر بگویم کتابخانه ها را برایمان نگه دارد.

دستور زیر یک سری فایل را برایمان در node\_package نصب میکند

npm install --save-dev jquery react react-dom webpack webpack-bundle-tracker babel-loader babel-core babel-preset-es2015 babel-preset-react

و این عبار ت --save-dev در دستور فوق باعث میشود که نام فایل های نصب شده به صورت اتوماتیک به package.json هم اضافه شود.

Package.json همان کاری که requirement.txt که pip می کرد را میکند. اما میتوان گفت این npm که به جای pip ان را ایجاد می کند.

کار redux چیست؟

State management یا به قولی manage کردن state های کامپوننت ها

از روی این اموزش که می روم:

<http://geezhawk.github.io/using-react-with-django-rest-framework>

خطایی میدهد که با این کد رفع می شود:

npm install webpack@2.1.0-beta.22 --save-dev

کد خروجی webpack :

این طور اتفاق افتاد که من هرچه کد موجود در index.jsx را تغییر میدادم هیچ تغییری در کد حاصل نمیشد. ابتدا گمان می کردم به علت کش کردن است رفتم موزیلا رو در قسمت پاک کردن سشن و کارهایی کردم اما فایده نکرد. سرور را با ctrl +c متوقف کرده دوباره ران کردم فایده نکرد رفتم داخل فولدر

G:\react\1\_from\_github\drf-react-master\assets\bundles\main-29fa6dbdcee8b71b95fc.js

را حذف کردم دیدم خطا میگیرد و هرچه دوباره کد را ران میکردم این بار خطا میگرفت که فایل فلان وجود ندارد. خلاصه دردسرت ندهم . دستور زیر را باید یاد بگیریم که هربار که کد را تغییر دادیم اجرا کنیم تا webpack برایمان کدهای جدیدی را ایجاد کند:

node\_modules\.bin\webpack --config webpack.config.js

این باعث تولید فایلی می شود که برای مرورگر قابل فهم است اینکه نام این فایل چیست و در کجا باید تولید شود را خودمان در webpack.config.js مشخص کرده ایم . در مثالی که من داشتم بررسی می کردم این خطوط در ان قرار دارند:

output: {

//where you want your compiled bundle to be stored

path: path.resolve('./assets/bundles/'),

//naming convention webpack should use for your files

filename: '[name]-[hash].js',

},

البته من هنوز نمیدانم این name و hash که در filename امده اند چطوری به چیزی مثل

main-29fa6dbdcee8b71b95fc.js

تبدیل شده اند.

باید ها و نباید های کدنویسی react :

ما نباید در متد render بیایییم state را تغییر دهیم.

State را باید با متد setState تغییر دهیم. پس نباید همچین کدی رو بنویسیم:  
this.state.term = event.target.value

ما در هر فایل فقط و فقط یک کامپوننت تعریف می کنیم

ایمپورت کردن فایل خروجی تولید شده توسط webpack :

در بعضی از اموزش ها دیده ام که فایلی مانند زیر را به صفحه ی html ایمپورت میکننند:

<script src=”bundle.js”></script>

کدی که من در جنگو دارم بدون این کار می کند پس فکر کنم جنگو خودش این را میداند و ایمپورت می کند.

کامپوننت ها را function base بنویسیم یا class base :

ما باید class base بنویسیم زیرا function base ها state ندارند.

فایل اصلی که نگهدارنده ی component هایی دیگر است:

ما فایلی داریم به نام webpack.config که تنظیمات را برایمان نگه میدارد. در این فایل صطری وجود دارد به این شکل :

entry: './assets/js/index',

این مولفه ی entry مشخث میکند که فایلی اصلی ما که کدهای react را داخلش نوشته ایم کجا قرار دارد و نامش چیست. اما فعلا نمیدانم که اگر ما بخواهیم چند app داشته باشیم ان موقع باید چیکار کنیم.

چرا نباید از jquery و react همزمان استفاده کنیم:

<http://tech.oyster.com/using-react-and-jquery-together/>

<https://stackoverflow.com/questions/38518278/how-to-use-jquery-on-reactjs>

می گویند spa از دکمه های عقب و جلو مرورگر پشتیبیانی نمی کند حالا علا این را داشته باشم تا ببینیم بعد چه می شود.

<https://developer.mozilla.org/en-US/docs/Web/API/History_API>